

ARTICLE OPEN

A scalable method for parameter-free simulation and validation of mechanistic cellular signal transduction network models

Jesper Romers^{1,3}, Sebastian Thieme^{1,3}, Ulrike Münzner^{1,2} and Marcus Krantz^{1*}

The metabolic modelling community has established the gold standard for bottom-up systems biology with reconstruction, validation and simulation of mechanistic genome-scale models. Similar methods have not been established for signal transduction networks, where the representation of complexes and internal states leads to scalability issues in both model formulation and execution. While rule- and agent-based methods allow efficient model definition and execution, respectively, model parametrisation introduces an additional layer of uncertainty due to the sparsity of reliably measured parameters. Here, we present a scalable method for parameter-free simulation of mechanistic signal transduction networks. It is based on rxncon and uses a bipartite Boolean logic with separate update rules for reactions and states. Using two generic update rules, we enable translation of any rxncon model into a unique Boolean model, which can be used for network validation and simulation—allowing the prediction of system-level function directly from molecular mechanistic data. Through scalable model definition and simulation, and the independence of quantitative parameters, it opens up for simulation and validation of mechanistic genome-scale models of signal transduction networks.

npj Systems Biology and Applications (2020)6:2; <https://doi.org/10.1038/s41540-019-0120-5>

INTRODUCTION

Systems biology aims at the integrative analysis of large-scale biological systems up to whole cells. To realise this goal, we integrate knowledge into executable or computational models.¹ This process has been developed the furthest in the field of metabolic modelling, where the community routinely works with genome-scale models. These models are defined at the level of biochemical reactions, cover the entire metabolic network of even complex cells, and can be simulated to predict system-level functionality.^{2,3} The methodology is well established and supported by rich toolboxes for network reconstruction, validation and simulation,⁴ and it constitutes the paradigm for bottom-up modelling. However, these tools cannot be used for signal transduction networks, due to the difference between mass and information transfer networks.⁵

Mechanistic modelling of signal transduction is challenging at several levels. First, at the level of model definition: empirical data on site-specific modifications of, or bonds between, signalling components combine combinatorially into a large number of possible configurations, or microstates.⁶ This combinatorial complexity effectively makes it impossible to create detailed mechanistic models of large signalling networks using models based on enumeration of microstates (as in e.g. normal ODE-models, Petri nets or SBGN-PD diagrams).⁷ Consequently, representation and simulation of these networks were limited to very small and/or heavily simplified models. To solve this, the community developed simulation tools with adaptive resolution, such as the rule-based modelling languages BioNetGen and Kappa.^{8,9} Second, simulation may be prohibitively expensive even with an efficient model definition, as is the case for classical rule-based modelling, in which the full network of microstates must be generated. This was solved by the development of the network-free simulation tool for

rule-based models, NFsim.¹⁰ Third, quantitative dynamic models require rate laws that must be parametrised in terms of rate constants and initial molecule amounts. Reliable information on these quantities is sparse, precluding meaningful parametrisation of most mechanistic models. While the lack of quantitative knowledge is an experimental rather than a theoretical challenge (more dedicated biochemistry is required), a simulation method that voids the need for parametrisation would be extremely helpful in evaluating mechanistic large-scale signalling models until this knowledge gap can be filled.

Parameter-free simulation of cellular networks is typically performed through constraint-based or Boolean methods.¹¹ Constraint-based methods rely on mass transfer *through* the network, and can therefore not be used to model signal transduction which primarily rely on interaction between and reversible modification of the signalling components. In contrast, Boolean methods can be used to simulate signal transduction networks. In these networks, entities (called “targets” below)—typically representing signalling components—are either true (active) or false (inactive) and edges define how the target states at time $t + 1$ depend on the target states at time t . Boolean networks have been used extensively in modelling, and different methods and toolboxes have been developed.¹² However, most of these Boolean models are not *mechanistic*, i.e. they abstract signalling to a binary on/off state of components, ignoring the actual state changes that transmit the information. While such phenomenological modelling have been used to simulate signalling successfully, models that account for the mechanisms of signal transduction increase the explanatory and predictive power by: (i) accounting for signalling components with more than two (on/off) states, such as the cyclin-dependent kinase that are activated against different targets by different cyclins. (ii)

¹Institute for Biology, Humboldt-Universität zu Berlin, Berlin, Germany. ²Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji, Japan. ³These authors contributed equally: Jesper Romers, Sebastian Thieme. *email: marcus.krantz@rxncon.org

Enabling the model to have the same resolution as data, facilitating model creation and evaluation as well as comparison to empirical data. (iii) Supporting functional analysis as e.g. in simulation of the effect of (combinations of) point mutations. Despite the potential advantages, only three mechanistic Boolean modelling methods have been developed until today: one is derived from SBGN-PD diagrams, one from rule-based models, and one is based on rxncon, the reaction-contingency language.^{13–15} These methods support a detailed description of signalling events. However, the first is based on a microstate description, inheriting the scalability issues of these approaches;¹³ the second requires a fully parametrised rule-based model, inheriting the problem of parametrisation;¹⁴ and the third inherited shortcomings in expressiveness and precision from the first generation of the rxncon language.¹⁵ Consequently, new methods are needed to support parameter-free, and hence scalable, simulation of signal transduction networks.

Here, we present a parameter-free simulation method that supports large-scale mechanistic models of signal transduction networks. This bipartite Boolean modelling (bbm) logic is based on the second generation rxncon language (Fig. 1), which is tailored for formalising signal transduction models based on empirical data:¹⁶ the reaction network is defined in terms of *elemental states*, i.e. modifications (or lack thereof) at specific residues and bonds (or lack thereof) at specific domains. The definition is bipartite: *elemental reactions* are decontextualised reaction events that define how elemental states are *synthesised*, *degraded*, *produced*, or *consumed*, and *contingencies* define how elemental reactions depend on (combinations of) elemental states that are not changed through the reaction. This structure is kept in the bbm: the *state targets* track which components are present, and in which states the components are (i.e., which domains are (not) bound to which other components and which residues are (un) modified). The *reaction targets* track which reactions are eligible to fire. Through a constructive approach, we define two generic update rules for these two target types, and demonstrated that the update rules can be assembled, LEGO-brick style, into large-scale models without any need for optimisation at the system level. Furthermore, we show that the resulting models meaningfully reproduce the behaviour of signalling processes, and that discrepancies between model behaviour and system-level expectations can be used to identify gaps in the network reconstruction and hence to improve the model. In a parallel effort, we use the rxncon language and the bbm formalism presented here to build and analyse a comprehensive and mechanistically detailed model of the network that controls the cell division cycle in baker's yeast¹⁷—validating both the scalability as well as the explanatory and predictive power of the method. Taken together, we present a scalable method for parameter-free validation of mechanistic signal transduction network models, taking an important step to close the gap in capabilities between metabolic and signal transduction modelling, and introduce a method for scalable simulation of signal transduction networks that supports modelling at the genome scale.

RESULTS

A new approach to Boolean modelling

The central result presented in this work is a method to derive a parameter-free Boolean model from a rxncon network, in a completely mechanistic fashion. The starting point is an arbitrary model defined in the second generation rxncon language, and the end point a bipartite Boolean model prepared for simulation with the Boolnet package.¹⁸ The model generation process is implemented in the rxncon compiler, which generates three output files: *A.boolnet* file containing the model, *a.csv* file containing the initial values

of all targets (see discussion below) and *a.csv* file linking the model aliases to the original rxncon names. The two first are used to simulate the model, and the third is necessary to interpret the results (see the methods section for detail). Hence, we use a well-established format for Boolean modelling, but the meaning we assign to nodes and edges are fundamentally different than those used in previous Boolean models. Below, we describe the derivation of this modelling logic and exemplify it on two minimal motifs, a small pathway and a detailed model of the yeast pheromone pathway that cannot be meaningfully simulated with conventional methods at this level of mechanistic resolution.

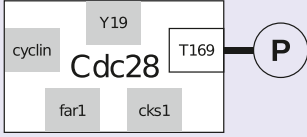
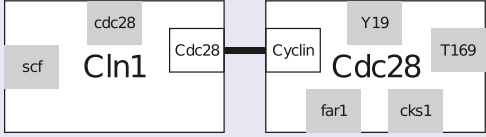
A rxncon model uniquely defines a bipartite network

A rxncon model can be represented by a bipartite network (elemental species-reaction graph, see Fig. 1), which consists of two distinct types of nodes: *elemental states* and *elemental reactions*. These nodes are connected by two distinct types of directed edges: *reaction edges*, which connect elemental reactions to elemental states, and *contingency edges*, which connect elemental states to elemental reactions. The mapping from a rxncon network to an elemental species-reaction graph is unique: the reaction list of the rxncon model contains the information needed to generate the nodes and the reaction edges, i.e. which elemental reactions and states that exist, which elemental reactions change which elemental states, and how. We consider two types of elemental states; *internal states*, which are the property of a single molecule (e.g. covalent modifications such as phosphorylation or ubiquitylation, or an empty binding domain) and *bond states*, which are shared between two molecules or two domains in the same molecule (e.g. protein-protein interactions and intra-protein interactions, respectively). Importantly, both these types of states are *elemental*, meaning they define a single, indivisible property of one molecule (two for intra-molecular bonds), at a single site (two for bonds) which we call residue (for modifications) or domain (for bonds). An elemental reaction can act on each state in one of four different ways: through *synthesis* (the state appears together with the component as the latter is synthesised), *degradation* (the state disappears together with the component as the latter is degraded), *production* (the state appears on an already present component) and *consumption* (the state disappears but the component remains). The mode of action is defined from the perspective of each state. For example, the Cak1_P + _Cdc28_[T169] phosphorylation reaction produces the phosphorylated Cdc28_[T169]-{P} state and consumes the unphosphorylated Cdc28_[T169]-{0} state (Fig. 1a, b). The effect of a reaction is directly defined by the *skeleton rule* underlying the reaction.¹⁶ This rule is similar to a rule-based model rule, such as can be defined in BNGL, but consists solely of a centre, without context:

The definitions, where RHS and LHS refer to the right-hand respectively left-hand side of the skeleton rule, are: *Production*: a state is produced by a reaction if it appears on the RHS, not on the LHS, but the component carrying the state does appear on the LHS. *Consumption*: a state is consumed by a reaction if it appears on the LHS, not on the RHS, but the component carrying the state does appear in the RHS. *Synthesis*: a state is

A Elemental States

Site-specific states; the typical empirical observable. Correspond to sets of microstates.

Type	Covalent modification	Bond
Locus resolution	Residue	Domain
Example	Cdc28 phosphorylated at residue T169 in the Tloop	Cdc28 bound to Cln1
rxncon notation	$\text{Cdc28_}[\text{Tloop(T169)}]\text{-}\{\text{P}\}$ Molecule: Cdc28 Locus: residue T169 in the Tloop domain State: phosphorylated	$\text{Cdc28_}[\text{cyclin}]\text{-}\text{Cln1_}[\text{cdc28}]$ Molecule: Cdc28 Locus: cyclin binding domain State: bound Molecule: Cln1 Locus: cdc28 binding domain
Neutral complement	$\text{Cdc28_}[\text{Tloop(T169)}]\text{-}\{\text{0}\}$ State: unmodified	$\text{Cdc28_}[\text{cyclin}]\text{-}\text{0}$ $\text{Cln1_}[\text{cdc28}]\text{-}\text{0}$ State: unbound
Cartoon		
# Microstates	16+	64+

B Elemental Reactions

Minimal reaction events changing elemental states. Correspond to sets of microstate reactions.

Example	Cak1 phosphorylates Cdc28 at residue T169 in the Tloop	The cyclin binding domain in Cdc28 binds (ppi+) the cdc28 binding domain in Cln1
rxncon notation	$\text{Cak1_P+_Cdc28_}[\text{Tloop(T169)}]$ Reaction: Phosphorylation (P+)	$\text{Cdc28_}[\text{cyclin}]\text{_ppi+_Cln1_}[\text{cdc28}]$ Reaction: protein-protein interaction (ppi)
reverse reaction	$\text{Ptc2_P-_Cdc28_}[\text{Tloop(T169)}]$	$\text{Cdc28_}[\text{cyclin}]\text{_ppi-_Cln1_}[\text{cdc28}]$
	Catalyst: Cak1 Target: T169 in Cdc28	Binding partner 1: cyclin binding domain in Cdc28 Binding partner 2: cdc28 binding domain in Cln1

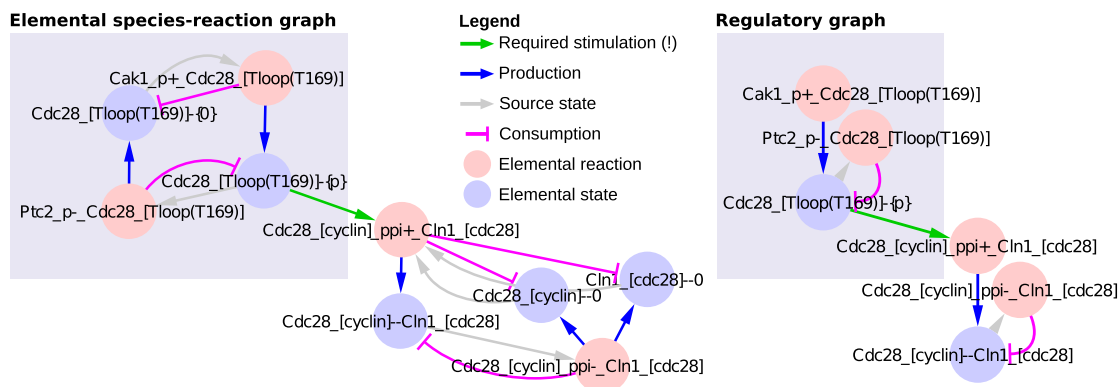
C Contingencies

Regulatory constraints. Specify the subsets of microstate reactions that fire.

Example	Cdc28 binding to Cln1 requires Cdc28 to be phosphorylated at T169 in the Tloop	
rxncon notation	$\text{Cdc28_}[\text{cyclin}]\text{_ppi+_Cln1_}[\text{cdc28}] \text{ ! } \text{Cdc28_}[\text{Tloop(T169)}]\text{-}\{\text{P}\}$	
	Target: Elemental reaction	Contingency: Absolute requirement Effector: Elemental state

D Visualisation

The bipartite reaction-contingency network is efficiently visualised in the regulatory graph.



synthesised by a reaction if it appears on the RHS, and the component carrying the state does not appear on the LHS. *Degradation*: a

state is degraded by a reaction if the component carrying the state appears on the LHS, no state mutually exclusive with it

Fig. 1 The reaction-contingency (rxncon) language employs three essential components. Elemental states, elemental reactions, and contingencies. The regulatory graph efficiently represents the regulatory structure of the network. **a** Elemental states are empirical observables: signalling molecules encode information through site-specific state changes, i.e. *covalent modification* of specific *residues* or *bonds* between specific *domains*. All elemental states at a single locus are mutually exclusive with each other and their neutral complements (unmodified and unbound, respectively). However, each elemental state only defines the state at a single locus (two for bonds; cartoon: white boxes), leading to a one-to-many relationship between the (empirically measured) elemental states and the (inferred) microstates. **b** Elemental reactions are indivisible reaction events defined in terms of elemental states. They are only defined in terms of the elemental states that are produced, consumed, synthesised or degraded, similar to the reaction centre of a rule in RBMs, and hence have a one-to-many relationship to microstate reactions. **c** The contingencies define constraints on an elemental reaction in terms of elemental states (or Inputs) that do not change through the reaction. Hence, the contingencies reduce the number of valid microstate reactions, and correspond to the reaction contexts of RBMs. Boolean contingencies, employing AND, OR and NOT gates, can specify arbitrarily detailed constraints down to microstates as necessary. Due to this adaptive resolution, rxncon models can faithfully mirror the empirical knowledge. **d** The regulatory graph (right) provides a compact representation of the regulatory structure of the rxncon network. It is a simplified version of the elemental species-reaction graph (left), leaving out the neutral states to reduce graph complexity, to remove non-informative cycles, and to improve readability. Both graphs are bipartite, with two types of nodes and two types of edges: reaction-to-state (reaction) edges show the effect of each reaction on its source and product states, and state-to-reaction (contingency and source state) edges the impact of states on reactions. The bipartite nature of the graph highlights the requirement for both reactions and contingencies for information transfer through the network, as both types of edges are necessary to traverse the graph. This figure and legend is reproduced from ref. ¹⁷.

appears on the LHS, and the component carrying the state does not appear on the RHS.

The contingency list contains the information needed to generate the contingency edges, i.e. which elemental states influence which elemental reaction, and how. This information corresponds to the reaction context of rule based models, i.e. they define the contextual constraints on the reactions in terms of elemental states that do not change through the reaction. There are six types of contingencies that define which role an elemental state plays in a reaction: absolutely required ("!"), stimulating ("K+"), no effect ("0"), inhibitory ("K-"), absolutely inhibitory ("x"), and no known effect ("?"; treated as "0"). The qualitative contingencies ("!/"/"x") must be fulfilled for a reaction to occur, while the quantitative contingencies ("K+""K-") change the rate with which a reaction occurs. Of course, many reactions depend on more than one elemental state, such as reactions requiring dual phosphorylation or the assembly of a multimeric complex. Such complex contingencies that require negation of and/or multiple elemental states can be defined using Boolean (AND, OR, NOT) contingencies, which, for clarity, are visualised as separate nodes in the graph.

Note: The resulting graph is deceptively similar to the bipartite species-reaction graphs used in e.g. metabolic networks or SBGN-PD diagrams. However, the elemental states are not disjoint—i.e., a single molecule typically occupies several elemental states. In fact, each molecule must occupy exactly one elemental state for each residue and domain that the molecule contains (see Fig. 1a). In contrast, SBGN-PD, the KEGG metabolic maps or Petri nets uses disjoint microstates that are mutually exclusive. Microstates are defined by a combinatorial enumeration of all (relevant) elemental states and therefore associated with severe scalability issues. In these, each molecule occupies exactly one microstate. This critical difference precludes the use of simulation methods developed for disjoint networks, and necessitates the development of a new simulation method tailored to rxncon network models.

From a rxncon network to a Boolean model: basic assumptions and desired model behaviour

The first step in defining the modelling logic is to define what true and false means for the elemental reaction and state targets. The interpretation we assign to the value of these targets is as follows:

If a *reaction target* is true, the cellular regulatory network is, at that point in time, in a configuration where it can accommodate that reaction. This is required, but not sufficient, for the reaction to

take place: In the absence of its source state(s), a reaction will not "fire" even though its value is true, as the reaction targets are purely a description of the regulatory layer of the biological cell. In rxncon terms, the reaction updates implement the contingencies.

A *state target* is true if there are a sufficient number of molecules carrying that state present in the cell for it to be considered functionally relevant. In rxncon terms, the state updates implement the elemental reactions.

The second step is to define the desired model behaviour. We desire both reaction and state targets to describe *system-level* properties, not molecular ones. A consequence of this is, for example, that two states that are mutually exclusive on a single molecule (e.g. a single residue being phosphorylated and unphosphorylated), can be simultaneously true in our Boolean system.

The behaviour of the reaction targets is determined by the contingencies. The basic interpretation is straightforward: reactions should only fire if all required ("!") and no absolutely inhibitory ("x") contingencies are fulfilled. We note that quantitative effects ("K+""K-") lack interpretation in a deterministic Boolean system, and chose to treat them as no effect ("0"), but also implement the option to treat them as absolutely required or inhibitory, respectively. To accommodate contingencies consisting of multiple elemental states we need to make our first assumption: that any combination of elemental states is true if all single elemental states are true. With this assumption, it becomes straightforward that for a reaction to be true, all required elemental states ("!") must be true and all absolutely inhibitory elemental states ("x") must be false.

The state targets are determined by the reactions. Here, we assume a quasi-steady-state at each update step. From this follows a natural hierarchy between the reaction types introduced above: *synthesis* is stronger than *degradation* which is stronger than *production* which is stronger than *consumption*. Due to the crude concept of Boolean time, we consider a quasi-steady-state at each update step. It is then easy to see that a protein that is both synthesised and degraded must be present in the system. In the case where the synthesis reaction is too weak to maintain functional level of the protein, it would be considered off. Similar, production, from the perspective of a specific state, will be dominant over consumption: in the presence of a phosphorylation cycle with both kinases and phosphatases active, both forms will be present. Finally, degradation is dominant over production, as depletion of a protein will deplete the phosphorylated form regardless of kinase activity (except when protected from degradation, as covered by contingencies).

To define the desired behaviour in detail, we designed and studied the minimal irreducible motifs containing two (for

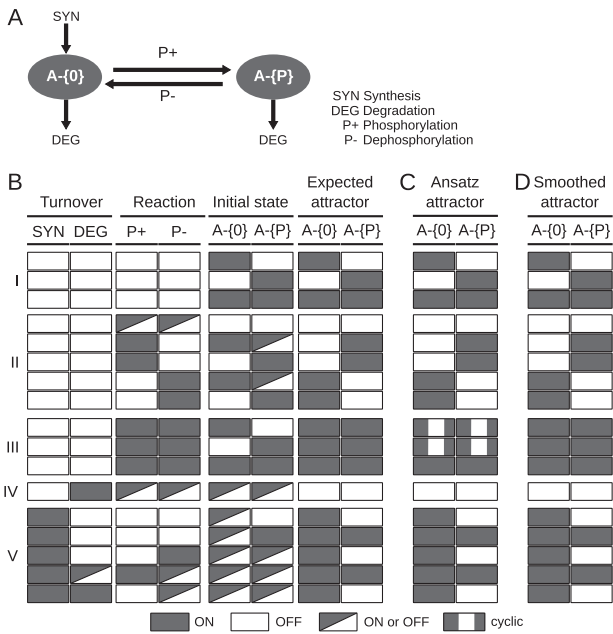


Fig. 2 Behaviour of a minimal modification motif. **a** The motif includes two states of A, unphosphorylated (A-{0}) and phosphorylated (A-{P}). The motif contains four different reaction types: component A can be synthesised (in its neutral state A-{0}), degraded (in either state), phosphorylated (consumes A-{0}, produces A-{P}) or dephosphorylated (consumes A-{P}, produces A-{0}). The regulatory and elemental species-reaction graphs of the motif can be found in Supplementary Fig. 1. **b–d** Initial conditions, expectations and simulation results. Each line corresponds to the simulation of one or more model variants, and visualise which reactions are active, which states are initiated, which behaviour we expect, and the behaviour we observed. **b** To define the desired behaviour of the motif, we create 64 variants of the motif with each of the four reactions constitutively ON (true) or OFF (false) (columns 1–4), and each of the elemental states initially true or false (columns 5 and 6), and define the expected steady state as a function of initial state and active reactions (“Expected attractor”; columns 7 and 8). (I) In the absence of any reactions, the steady state will be identical to the initial state. (II–III) In the absence of synthesis or degradation, but in the presence of component A (A-{0} or A-{P} true), the equilibrium depends on the (de)phosphorylation reactions. With only one of these reactions, only the fully (de)phosphorylated form is present at steady state. However, with both reactions present we expect both elemental states to be present at steady state. (IV) With degradation but not synthesis active, the protein will be depleted and both states will be false at steady state. (V) With active synthesis, the neutral state will always be present. The phosphorylated state will only be present if there is also a phosphorylation reaction or if the state is initially present and both the degradation and dephosphorylation reactions are off. **c** The attractors reached after simulation with the update rules in the original ansatz. The attractors correspond to the expected attractors (see B) for 62 out of 64 configurations. The exception are the cyclic attractors in block (III), where both phosphorylation and dephosphorylation are active and only one of the elemental states are initiated as true. **d** The attractors reached after simulation with the smoothed update rules. The attractors are identical to the expected attractors in all cases.

modification reactions; Fig. 2a) or three elemental states (for interactions; Fig. 3a), and the reactions acting upon them. Each motif contains four reaction types that synthesise, degrade, produce or consume the states. By varying which reactions are active and inactive, and which states are initially present, we arrive at 64 (2^6) and 128 (2^7), respectively, distinct models for the two motifs. We define, given an initial configuration for the states and the absence or presence of each of the reactions, the desired steady-state behaviour (Figs. 2b and 3b). The crux of the matter

then becomes to find update rules for the states so that the (deterministic) attractor reproduces this steady state.

Defining the notation: reactions, states and components

A rxncon system will contain N_R reactions denoted by R_i , N_S elemental states denoted by S_i , and N_C components denoted by C_i . For the components that appear without any internal states, such as e.g. those that solely appear as catalysts, the component is its own state (which does not appear in the original rxncon system). For those components that do carry internal or bond states, the component can be expanded as a Boolean expression of elemental states grouped by the site (domain or residue) on which they live (Eq. 1):

$$C_i = \bigcap_{\text{site } j \text{ on component } i} \bigcup_{\text{state } k \text{ on site } j} S_k \quad (1)$$

The origin of this expression is in the mutual exclusivity of elemental states that live on the same residue or domain: for each of these sites, at least one of the states living on the site needs to be present for the component itself to be present. A consequence of this is that for components that carry internal or bond states, the dynamic behaviour of the component is fully determined by the states which the component carries.

Furthermore, we define functions mapping states and reactions to Boolean expressions of states. First, the functions $K(R_i)$ list the components which are reacting in reaction R_i (Eq. 2):

$$K(R_i) = \bigcap_{\text{component } j \text{ reacts in } R_i} C_j \quad (2)$$

whereas the functions $K(S_i)$ list the components carrying the state S_i . Bond states are carried by two components, whereas modifications are carried by one (Eq. 3):

$$K(S_i) = \bigcap_{\text{component } j \text{ carries } S_i} C_j \quad (3)$$

In the update rule for the states, we will require the following combinations. First, a reaction together with its source states (Eq. 4):

$$R_i = R_i \bigcap_{\substack{S_j \text{ consumed} \\ \text{by } R_i}} S_j \quad (4)$$

The neutral state (unmodified, unbound) counterpart for a particular state S_i is denoted by $N(S_i)$.

These notations enable us to write down the synthesis term Σ , which describes whether a state is either directly or indirectly being synthesised (Eq. 5):

$$\Sigma(S_i) = \begin{cases} \bigcup_{R_j \text{ synthesizes } S_i} R_j & \text{for neutral states } S_i \\ \bigcup_{R_j \text{ synthesizes } N(S_i)} R_j \cap \bigcup_{R_k \text{ produces } S_i} R_k & \text{for non-neutral states } S_i \end{cases} \quad (5)$$

All systems we considered contain only states “one step” removed from the neutral states, so the expression for non-neutral states describes an active path coming from the synthesised state to the state under consideration. For states that are multiple steps removed from the synthesised neutral state, this expression has to be appropriately amended.

Finally, we denote the Boolean expression representing the contingency for reaction R_i by $L(R_i)$ (Eq. 6):

$$L(R_i) = \bigcap_j L_j^1(R_i) \bigcap_k \overline{L_k^x(R_i)} \quad (6)$$

where the $L_j^1(R_i)$ and $L_k^x(R_i)$ enumerate the required, respectively, inhibitory contingencies for reaction R_i , which are themselves possibly nested Boolean expressions.

The expected behaviour of a small reaction circuit and update rule ansatz. The reaction update rules are quite straightforward. We require the strict contingencies for the reaction to be satisfied and the presence of the components on which the reaction acts (Eq. 7):

$$R_i(t+1) = K(R_i; t) \cap L(R_i; t) \quad (7)$$

The state update rules are more complex and explicitly require the hierarchy between types of reactions that was alluded to above. First of all, if a state is synthesised by any reaction, it will be true. If synthesis is false, the necessary, but not sufficient, requirement for the state to be true is that degradation is also false, and that the component(s) carrying the state are present. Now, there are two options for the state to be true: either the state is being produced by some reaction, in which case it is immaterial what the previous value of the state was, or the state was already true and it is not being consumed by any reaction (Figs. 2 and 3). Eq. 8:

$$S_i(t+1) = \Sigma(t) \bigcup \left(K(S_i; t) \bigcap_{R_k \text{ degrades } S_i} \overline{R_k^x(t)} \bigcap \left\{ R_i \text{ produces } S_i \bigcup_{R_m \text{ consumes } S_i} \overline{R_m^x(t)} \right\} \right) \quad (8)$$

In other words, a state remains true as long as it is neither consumed nor degraded. If a reaction produces the state, it becomes or remains true even when it is actively consumed, but not if it is degraded. Finally, if the state is synthesised, it becomes or remains true regardless of any other active reaction. See supplementary discussion for an example.

As can be seen in this formula, the translation of “the state is being produced” *et cetera* contains the primed reactions, meaning the reaction producing that state and the source state(s) of that reaction (see Eq. 4). This is due to the semantics of the reaction targets, which only tell us about the regulatory state of the network.

Note that most reactions act on multiple states (typically source and product states). It is imperative that these are updated simultaneously to avoid simulation artefacts. Hence, we use only synchronous deterministic update schemes.

Testing the generic update rules

To test if the update logic described above captures the expected behaviour, we implemented the model generation process and used it to generate the 64 models corresponding to the minimal modification circuitry above (Fig. 2). The models were simulated using BoolNet,¹⁸ as described in the methods. The attractor states are visualised in Fig. 2c. The model behaviours correspond to our expectations with one notable exception. In the absence of synthesis and degradation, but in the presence of both phosphorylation and dephosphorylation, the model displays an oscillatory behaviour when only one of the two states is initiated. Closer inspection reveals that this is due to periodic source state depletion. The phosphorylation and dephosphorylation reactions are constitutive (no contingencies, no loss of components), and the oscillations occur due to the state updates, which are

completely encoded in the state update rule. Indeed, as soon as a reaction executes, dependent on its source state, it depletes the source state pool. Hence, the reactions alternate in firing, triggering out-of-phase oscillations in the truth value of the states. Consistently, these oscillations disappear when both states are initiated or when the source state is replenished through synthesis. We observe the same phenomenon in the interaction motif, when the reaction cycle is active and at least one component is initiated in and remains in a single form (Fig. 3c). We consider these spurious oscillations undesirable in our systems-level description, but note that they would be appropriate for models of single molecules. Nevertheless, the outcome is highly encouraging, as 62 out of 64 models matched the expected behaviour.

Source state smoothing eliminates the spurious oscillations

To eliminate the oscillations that plagued our initial ansatz, we adapted the updates rules for states by widening the window in which we checked for source state availability: a reaction needs the source state to be present, or to be produced. To achieve this, in Eq. (8) we substitute for the reactions producing the state, R_i' the following (Eq. 9):

$$R_i'(t) = R_i(t) \bigcap_{\substack{S_j \text{ consumed} \\ \text{by } R_i}} S_j(t) \bigcup S_j(t+1) \quad (9)$$

where the $S_j(t+1)$ is the full expression (Eq. 8), without this substitution.

We consider this adaptation quite natural: there are large numbers of molecules undergoing the same set of reactions. Therefore, it is highly unlikely that all of these reactions are temporally completely in phase, justifying a smoothing over molecules. In addition, the time scale in a Boolean model is basically set by the slowest of the reactions, since all rate constants are absent. For molecule pools acted upon by reactions that are faster than the slowest in the system, it is likely that they will pass through mutually exclusive states within the window of a Boolean time step, justifying a “time smoothing”. Both effects are captured by the reworked update rules.

The smoothing assumption only breaks down in the context of few molecules and low reaction rates, and there are cases in which smoothing is inappropriate. In other work,¹⁷ we have introduced a hybrid model containing both the molecular reactions and states described here, and additionally macroscopic reactions and states that are governed by the non-smoothed update rules. However, for most states in a signal transduction network, the smoothed update rule is more appropriate. Having established the smoothing logic, we implemented it into the rxncon compiler tool and recreated the 64 modification motif (Fig. 2) and 128 interaction motif (Fig. 3) models with smoothing. We repeated the simulation and compared the results to the original simulation (Fig. 2d, Fig. 3d). The oscillatory behaviour disappeared, but no other simulation results changed. Hence, the simulation results exactly match the behaviour we expect from a model of these reaction motifs.

The update rules can be used as LEGO bricks to assemble a systems level model

Next, we applied the bBM logic to simulate a linear pathway. Both the reaction and contingency motifs are fully defined at the local level, as they only depend on the reactions (for state updates) and the contingencies (for reaction updates) in the system. We postulated that these locally defined update rules would suffice to define system-level function, and tested this hypothesis with a simplified model of the HOG MAP kinase pathway from *Saccharomyces cerevisiae* (Fig. 4; adapted from¹⁵). We created a rxncon 2.0 model of this pathway (Supplementary

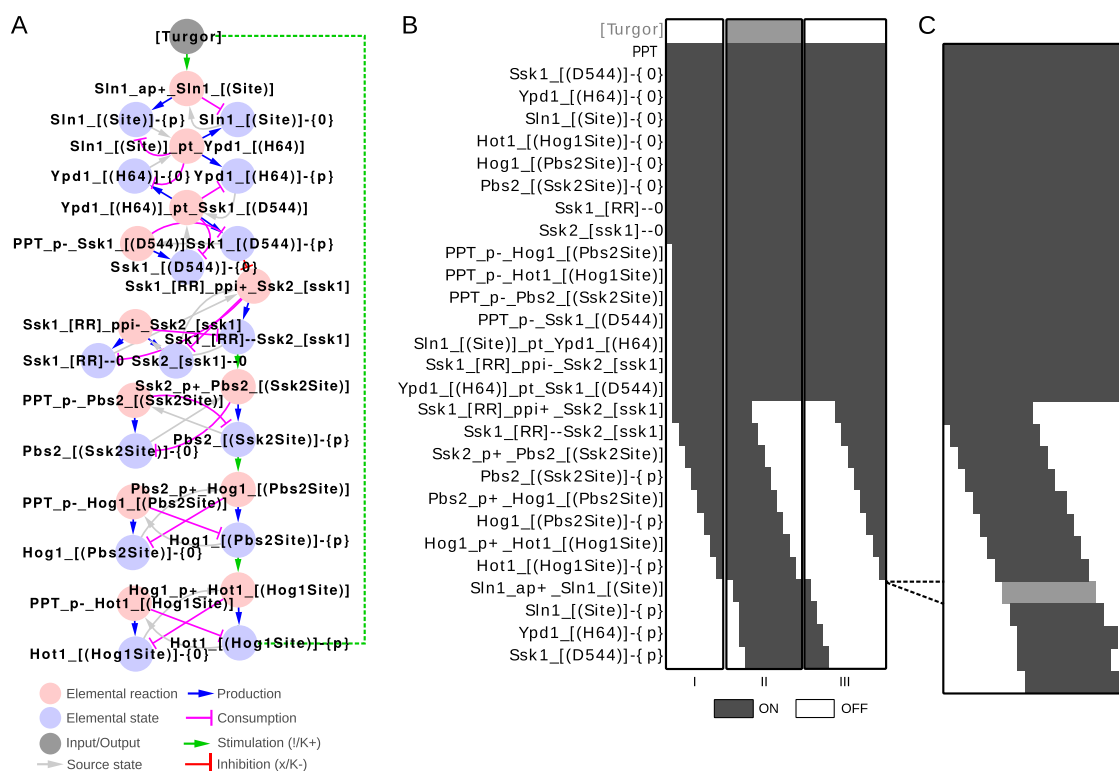


Fig. 4 From reactions to a functional pathway. We used the smoothed update rules to generate and simulate a model of the Sln1 branch of the high osmolarity glycerol (HOG) pathway. **a** The pathway visualised as a rxncon regulatory graph. In the absence of turgor, Sln1 stays unphosphorylated. As turgor increases, the auto-phosphorylation of Sln1 initiates a phosphotransfer cascade converging at Ssk1. The phosphorylated form of Ssk1 turns off the downstream MAP kinase pathway leading to dephosphorylation of the downstream transcription factor Hot1. The dashed line indicates a feedback loop that is included in the cyclic version only. The model has 29 different targets: 12 reaction targets (pink nodes), 16 state targets corresponding to 15 elemental states (blue nodes) and a component state for the phosphatase (not displayed in the graph), and the single input/output target “Turgor” (grey node). **b** Simulation of a linear version of the model using source state smoothing of the update rules. (I) We use our default assumptions on the initial state and simulate the model until we reach an attractor (first OFF trajectory). (II) We activate the system by turning [Turgor] ON and simulate again (ON trajectory) until we reach an attractor state. (III) From there, we set [Turgor] OFF again and simulate the model until we reach an attractor. We observe that the model responds as expected to the input. **c** We extended the HOG model with a feedback loop, where activation of the pathway leads to increased turgor (via Hot1-P). This is simplification of an adaptive response through increased glycerol production and retention, which increases turgor. We simulate this model from the initial OFF attractor (see panel B), and note that the system oscillates as expected: the trajectory is cyclic, where the last time step is followed by the first, and turgor is now a model variable that turns on and off during the cycle (grey row in the heatmap).

Model 1), and used this to generate the bBM using the generic update rules with smoothing (see Supplementary discussion for implementation details). Already this small model has 28 reaction and state, and hence 2^{28} ($\sim 10^8$) possible initial states. We deemed this too many for an exhaustive search, and decided to use a generic start state for all simulations: all neutral elemental states (that is, unbound binding domains and neutral modifications) are true, all generic component states (for components with no elemental states) are true, and all other targets are false. From this highly artificial initial state, we let the model find its own natural “off state” by executing it until an attractor is reached (Fig. 4b). At this point, we take this attractor as a new initial state and change the input state (Turgor in this case) and repeat to see the response of the pathway to the input, and repeat this process until the model returns to a state vector we have already seen. As can be seen from Fig. 4b, the HOG pathway responds appropriately to turgor: it turns off the kinase cascade. For comparison, we repeated the simulation with the non-smoothed logic (Fig. 5a), where we see the signal passing through the network despite spurious oscillations. However, the system does not converge to point attractors, leading to more complex analysis and interpretation. There are three striking blocks in the heatmap (Fig. 4b): first, the initial neutral states never turn off. Second, there is a block of reactions that turns on directly, and stays on throughout the simulation.

Third, there is a block that turns on and off in response to the signal. The third block contains the reactions and states that actually transmit the information. The second block contains constitutive reactions, which are either unregulated (e.g. dephosphorylation reactions), or regulated at the level of source state availability (e.g. phosphotransfer from Sln1 to Ypd1). The first block contains all the neutral states. These remain true because the reactions that produce them are considered unregulated, which may be due to experimental bias as discussed below. Hence, the logic of the generic update rules is sufficient to convert the molecular level knowledge in a rxncon network into a functional bBM that accurately predicts system-level function. It is highly non-trivial that generic update rules, which were defined for isolated reactions and states, suffice to define a complete model that functions at the systems level, with no further tweaking or parametrisation. Taken together, the generic update rules map a given rxncon network on a unique Boolean model that predicts systems level function.

The bipartite Boolean logic correctly reproduces real oscillations. The HOG pathway is a homeostatic pathway that maintains proper turgor pressure. The pathway output eventually leads to signal cessation through a physiological feedback loop.¹⁹ To simulate this, we linked the most downstream component to the input that

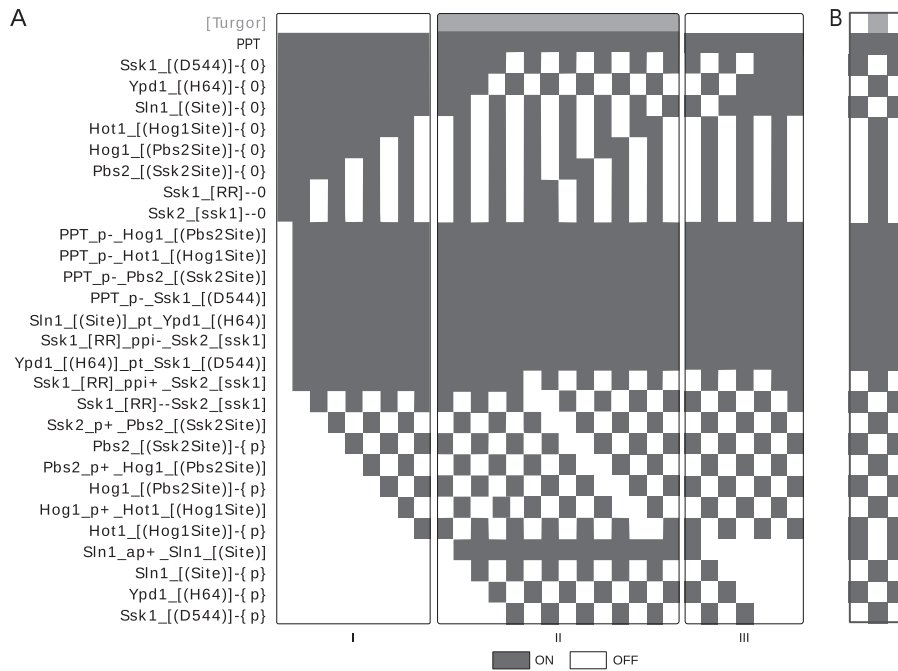


Fig. 5 Smoothing is required to make pathway simulation interpretable. We repeated the simulation of the HOG pathway with the unsmoothed update rules from the initial ansatz. **a** Simulation of the linear model without smoothing. The signal goes through the pathway, but analysis is complicated by the spurious oscillations as the system no longer converges on point attractors. Each of the three simulation trajectories (I–III) ends with a cyclic attractor of length two. **b** Simulation of the cyclic HOG model without smoothing. Here, the entire oscillation cycle breaks down into a period two oscillator involving all the states (and their complements) and reactions that transmit the information.

turns the pathway off (dashed line in Fig. 4a). We repeated the model creation and simulation, using the initial steady state of the linear model as starting condition. As shown in Fig. 4c, the model now shows a periodic activation/deactivation behaviour, similar to that when the input is changed manually. Hence, the bBM logic is fully capable of predicting biologically relevant oscillations. As comparison, we also simulated the cyclic HOG model without source state smoothing (Fig. 5b). Here, the pathway signal is completely washed out by the spurious oscillations and breaks down to a two-state cyclic attractor. The source state smoothing facilitates bBM analysis and clearly improves the interpretability of the simulation results. Taken together, the bBM logic generates Boolean models that can predict systems level function for both linear and cyclic systems.

The bipartite Boolean logic scales to large-scale systems

Finally, we applied the method on the pheromone response pathway of baker's yeast. We chose this pathway to benchmark the bBM method due to the existence of an excellently annotated, comprehensive and mechanistically detailed rule based model (RBM).²⁰ The original RBM contains 229 rules with 200 parameters (166 unknown) that define how 18 components can assume over 200,000 distinct states (http://yeastpheromonemodel.org/wiki/Extracting_the_model). While this is one of the most carefully built and curated RBMs, it remains difficult to meaningfully simulate it as such.²¹ Hence, it constitutes an excellent benchmark target for the bBM method.

We simulated the pheromone bBM using a standardised simulation workflow.²² The rxncon translation of the RBM, which is described elsewhere,¹⁶ is defined by 95 elemental reactions and 231 (non-zero) contingencies. We generated the bBM using the smoothed update rules, which produced a bipartite Boolean model with 130 reaction targets (due to separation of bidirectional reactions and duplication of degradation reactions) and 118 state targets. With 248 targets, the model is too large to

use an exhaustive search of initial states (statespace = 2^{248} ; ca 10^{74} distinct configurations), so we fall back on our default initiation state vector (all neutral state targets are true, all generic component targets true, all other targets are false). From this initial state, we first simulated the model to a first point attractor, to let it find its natural “off state”, as explained for the HOG pathway above: thereafter, we iteratively switched the input to true and false. Analysing the outcome, we found that the pathway was constitutively active and unresponsive to pheromone. First, we examined if this was due to the interpretation of quantitative effects that are lost in the Boolean model. However, neither treating all quantitative (“K+”/“K–”) contingencies as absolute (“!”/“x”), nor treating them as no effect (“0”), solves the problem. Furthermore, the original RBM was never simulated and proven to be functional. Hence, we proceeded with the minimal model (treating quantitative contingencies as no effect) and looked deeper into the pathway behaviour, finding that it activates in the absence of signal due to constitutive release of Ste4, which represents the beta/gamma subunit of the trimeric G-protein at the top of the cascade, as well as constitutively free—and hence active—Ste12. To address these problems, we change four quantitative contingencies into qualitative contingencies. In addition, we needed to limit turnover of Ste4 bound Gpa1 (to prevent signal-independent release of unbound Ste4, the activator of the pathway) and to remove Fus3 dependent degradation of Ste12 which made the pathway “single-shot”. The final model with changes can be found in Supplementary Model 2, and the simulation trajectories are shown in Fig. 6. This updated version of the model responds to pheromone exposure and withdrawal as expected, despite containing 83 quantitative contingencies—which are ignored in the model generation—indicating that a much simpler model would suffice to capture the key features of the pathway.

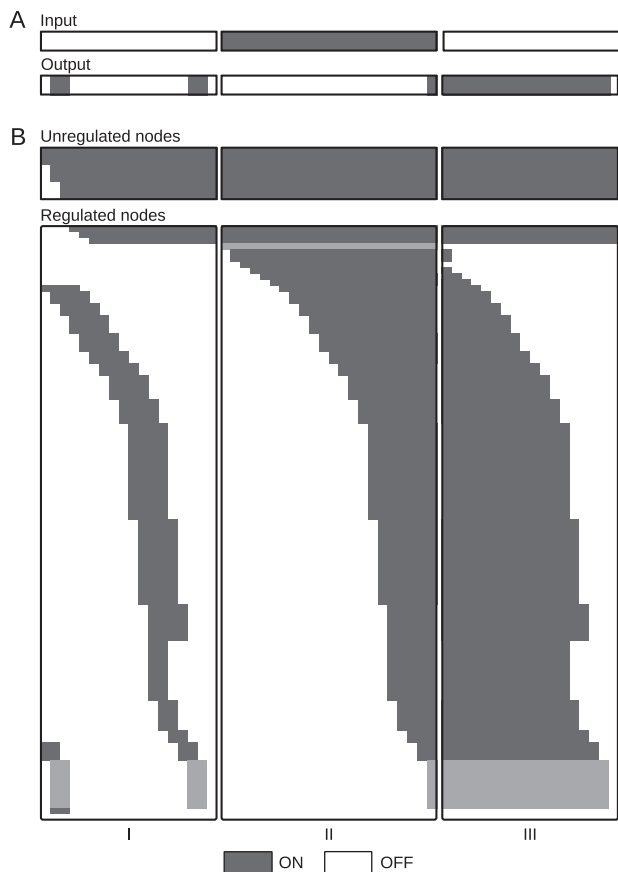


Fig. 6 The updated pheromone model responds as expected to pheromone. We generated a bBM from the updated pheromone model and (I) simulated it in the absence of pheromone (unbound pheromone set to false) until the first steady state, where (II) free pheromone was set to true, representing pheromone stimulation, and the simulation repeated until next steady state was reached, before (III) pheromone was removed (by setting both free and bound pheromone to false) and the model simulated to the next steady state. The pathway turns on and off as expected, and finds a natural off state in the first simulation despite two activation pulses that go through the pathway. These are due to proteins that activate the pathway in their neutral states: the upstream Ste4, as discussed in the text, and the Ste12 transcription factor—which, according to the model, is active when not bound to Dig1 and Dig2. However, both pulses are transient and the steady state is robust against these transient dynamics. Panel (a) displays the input and output trajectories and panel (b) all the 248 state and reaction trajectories with the 149 unregulated targets condensed to three lines. Grey lines in (b) indicate the input/output states as displayed in (a).

DISCUSSION

Here, we present a qualitative simulation method for large-scale, mechanistically detailed signal transduction network models. The formalism is based on Boolean logic and can be simulated and studied by a standard package such as BoolNet. However, we present a fundamentally new concept to formulate Boolean models. First, we create a bipartite model at the level of elemental reactions and states, capturing the key elements in signal transduction at an appropriate resolution for mechanistic modeling of these processes. Second, based on detailed analysis of two minimal reaction motifs, and on a small set of standard assumptions, we define two generic update rules: one for reaction targets and one for state targets. These generic update rules map a bipartite rxncon network on a unique bBM with defined truth tables. The elemental reactions define the update rules for the state targets, and the contingencies define the update rules for

the reaction targets. We show that these building blocks can be assembled like LEGO bricks into a bipartite Boolean model that predicts system-level function from molecular mechanisms, without optimisation at the system level.

The unique mapping from rxncon to an executable bBM that predicts system behaviour is highly non-trivial. Normally, it is relatively easy to build a Boolean model structure, but challenging to define truth tables that enable the model to reproduce the behaviour of the system.²³ Here, we find that the regulatory structure encoded in the rxncon network already uniquely defines a Boolean model with set truth tables, and that this Boolean model meaningfully predicts system-level behaviour. Thus, the bBM logic we present here bridges the microscopic (biochemical reactions) and macroscopic (input-output) levels of cellular signal transduction, fulfilling the requirements for the cellular “mechanics” proposed by Hlavacek and Faeder—at least qualitatively.⁶

This has far-reaching implications: first, it provides an efficient validation tool in the model building process. This allows the model construction to be separated in to two phases: a qualitative and a quantitative phase. Boolean models are computationally inexpensive, and the automatic model generation supports iterative model creation, analysis and improvement. In addition, we are better equipped with knowledge at the qualitative level, suggesting that this level should be optimised first. As rxncon supports compilation into both RBMs and bBMs (as well as several graphical formats), it can be used to facilitate this process:²² the structural model can be created and validated using graphical tools and bBM simulation, and later the improved network can be used to create a rule-based model. Hence, the more expensive parameterisation cycles can be performed after the qualitative model has passed the validation process. Second, the bBM method can be used for validation of large-scale signal transduction networks. Previously, large-scale reconstruction of signal transduction has been limited to graphical maps that cannot be executed.^{24–26} The method we present here changes this: we can now validate—through simulation—large-scale reconstructions of signal transduction networks.

The bBM formalism we presented here is fundamentally different from its previous incarnation.¹⁵ First, it has been designed to capture system-level behaviour, not the states of individual molecule instances in the network, meaning that states that would be mutually exclusive on a single molecule can be true at the same time. As we show in Fig. 5, this is critical for meaningful system-level predictions. Second, we used a constructive approach, defining all possible behaviours at the level of two families of minimal reaction motifs, to design two generic update rules. These update rules can be used to map any rxncon system, even extended by new reaction types, through the interpretation of the flexible skeleton rule definition as synthesis, degradation, production or consumption of different states. Third, the method we present here inherits the expressiveness and flexibility of rxncon 2.0, including the explicit representation of neutral states.¹⁶ Fourth, the reimplementation has improved the model generation, enabled the use of different export options, and improved the model creation and analysis workflow. While the previous incarnation worked well in many instances,^{15,27,28} these models had issues with certain reaction types (most notably degradation) and spurious oscillations. That the latter appeared so rarely was due to the implicit dominance of modified states: neutral states were not explicitly represented, making modification or binding reactions dominant over reactions that returned components to their neutral state. Here, we eliminate this artificial hierarchy, which we consider undesirable, and explicitly include the neutral states. However, most of these states are constantly true in our simulations. This may have two reasons: first, it could reflect biology: there would be a constant pool of unmodified components as long as there is a constant turnover (and hence synthesis, which per definition occurs in the neutral states). Second, it could reflect an experimental bias, as we know much more about the modifying

reactions than about the reactions that reverse the modification (e.g. phosphorylation vs dephosphorylation²⁹). If so, the formalism we present here helps us make this information bias explicit, and will allow us to integrate the regulation on these reactions as the knowledge becomes available.

Finally, the method enables a scale-shift in signal transduction modelling. Hitherto, executable signalling models have been mechanistically detailed or large-scale, but not both. Most mechanistic large-scale reconstructions are technically microstate models that could be simulated after parametrisation. However, they are actually divided into several unconnected modules and could hence not be simulated at the system level.^{24–26} Rule-based modelling languages have been used to build relatively large models,^{27,30} but even these models are limited to few (18 in these cases) components and parametrisation is already an outstanding challenge. In contrast, comprehensive signalling models will need to account for hundreds or thousands of signalling components, carrying many thousands of distinct elemental states. Here, we present a method that can deal with mechanistic signalling networks at this scope, and we have successfully used this method to build and analyse a comprehensive mechanistic model of the yeast cell division cycle, which accounts for 229 proteins, 790 elemental reactions and 1238 elemental states down to residue resolution when applicable¹⁷—far beyond the potential of previous mechanistic modelling. The method is qualitative, but this may be an advantage given the sparsity of reliable quantitative information on rate constants. In addition, even metabolic modelling—clearly the state of the art in genome-scale modelling—is limited to qualitative or semi-quantitative simulation methods at the genome scale.³¹

Taken together, we present a parameter-free model creation and simulation method for models of signal transduction. Unique models are generated directly from empirical data formalised in the rxncon 2.0 language, without need for fitting or optimisation. For the first time, we can simulate mechanistic models of signal transduction network at the genome scale.

METHODS

rxncon installation and execution

The rxncon framework requires Python 3.5 or 3.6. Make sure you have one of these Python versions installed. Anaconda (<https://www.continuum.io/downloads>) provides an easy way to install the most current Python version. With Python installed and up to date, you are ready to install rxncon:

Under Windows:

1. Open the console and type “pip install rxncon”.¹ The default installation folder will depend on your Python installation. With Anaconda, the rxncon folder appears in [user]/Anaconda3/lib/Site-packages. The files you will need to call appear in [user]/Anaconda3/Scripts.
2. To test the installation, navigate the console to the folder with the scripts and type “python rxncon2bngl.py”.² Expect a string “Usage: rxncon2bngl.py [OPTIONS] EXCEL_FILE” and an error message “Error: Missing argument “excel_file””.

Under OS X:

1. Open the console and type “pip install rxncon”. The default installation folder will depend on your Python installation. With Anaconda, the rxncon folder appears in [user]/Anaconda3/lib/python3.6/Site-packages. The files you will need to call appear in [user]/Anaconda3/bin.
2. To test the installation, navigate the console to the folder with the scripts and type “python rxncon2bngl.py”.³ Expect a string “Usage:

```
rxncon2bngl.py [OPTIONS] EXCEL_FILE” and an error message “Error:
Missing argument “excel_file”.
```

Under Linux:

1. Make sure you have PIP installed. If not, use your package manager to install it. E.g., on debian-based systems type “sudo apt install python3-pip”.
2. Open a terminal and type “pip3 install rxncon –user”. This installs into \$HOME/.local, the executables are in \$HOME/.local/bin.
3. To get easy access to the rxncon scripts, you can update your PATH environment variable to include this directory: put something like “export PATH=\$HOME/.local/bin:\$PATH” into your .bashrc.
4. To test the installation, type “rxncon2bngl.py”.⁴ Expect a string “Usage: rxncon2bngl.py [OPTIONS] EXCEL_FILE” and an error message “Error: Missing argument “excel_file””.

BoolNet installation. The logical simulation of rxncon networks uses BoolNet, an R package. To use these tools:

1. (Optional) Download and install R-studio (<https://www.rstudio.com>).⁵
2. Make sure you have R installed. R can be installed through Anaconda, by opening the console and typing: “conda install –c r r-essentials”.⁶
3. The BoolNet package can be installed from R. In the console, type “R” to enter the R environment. Then type “install.packages (“BoolNet”)” and select the download server.

Model creation and analysis. The creation of the rxncon models are described elsewhere. The High Osmolarity Glycerol (HOG) model was taken from¹⁵ and adapted to rxncon 2.0. The pheromone (PHER) model was translated from the yeastpheromonemodel.org wiki as described in.¹⁶ The bipartite Boolean model files were created with the rxncon compiler software, by calling the “rxncon2boolnet.py” script on the model (.xls) files with default setting:

```
python rxncon2boolnet.py path/model.xls
```

from the folder where the rxncon3.boolnet.py script is located, and where path/model.xls is the path to and file name of the rxncon model, with the file extension.

To access the possible options, call the script without a but with the –help command:

```
python rxncon2boolnet.py help
```

This lists the possible options that can be appended to the call command, e.g:

```
python rxncon2boolnet.py path/model.xls
k_plus strict k_minus strict
```

to run the model generation considering with quantitative contingencies (“K+”/“K–”) considered absolute (“!”/“x”) instead of being ignored.

The rxncon2boolnet.py script generates three files. First, the bipartite Boolean model file (<ModelName>.boolnet) contains the update rules using states and reaction IDs. Second, the symbol mapping file (<ModelName>_symbols.csv) defines which IDs correspond to which states and reactions in the rxncon file. Third, the initial vector (<ModelName>_initial_vals.csv) sets the initial state of the Boolean simulation.

Model simulation was done with the R CRAN package BoolNet.¹⁸ To facilitate simulation, we prepared an R script that can be downloaded from

⁴This only works with the path set. Without this, type “=\$HOME/.local/bin/rxncon2bngl.py”.

⁵Depending on installation method, this may or may not come with R. If not, the location of R must be set in the dialogue box. If R is installed from Anaconda, the windows path would be “[user]/Anaconda3/R”.

⁶If the conda command is not recognised, try to close and reopen the console.

¹On some computers, the installation of Pyeda fails for unknown reasons. In this case, try typing “pip install—no-cache-dir rxncon”.

²In this case, by typing “cd Anaconda3/Scripts”.

³In this case, by typing “cd Anaconda3/bin”.

<https://github.com/rxncon/tools> (BoolNetSim.R). To use this script through R studio:

- Save the network files and the R script into a single directory.
- Start RStudio.
- Open a new project and create it in the directory where you saved your files.
- Make sure your model files are located in the project folder.
- Open the R script. Set the filePrefix in the R script to <model>.
- Execute the entire script by selecting all text (ctrl+a) and pressing ctrl+enter.

The script generates five files: (i) <ModelName>.pdf, which graphically displays the simulation trajectory from initial state to the attractor, (ii) <ModelName>_trajectory_first.csv, with the trajectory as values (0/1) in tabular format, (iii) <ModelName>_2.pdf, which graphically displays the simulation trajectory from the attractor (useful to distinguish a point attractor (two columns) from a cyclic attractor (>2 columns)), (iv) <ModelName>_trajectory_second.csv, with the second trajectory in tabular format, and (v) <ModelName>_new_attractor.csv, with the new attractor as an initial values file.

Where "<ModelName>" is the file name (without extension) of your rxncon model.

Within a Boolean model, we expect the output to be responsive to the input. The sign of the dependence does not matter, and we can start with the input either on or off. We simulate the model until it reaches an attractor. If it is a point attractor, we use it as starting point for the next simulation but turn the Input signal into a truth value, activating the output target and simulate again until we reach another attractor. We iteratively change inputs and simulate to an attractor state until we reach an attractor we have already seen. For a detailed discussion of the workflow, see²² (pre-print at arXiv: <https://arxiv.org/abs/1802.01328>).

Software and model availability. The rxncon software is open source, distributed under the IGPL licence, and can be installed from the python package index with "pip install rxncon" (code also available at <https://github.com/rxncon/rxncon> but without dependencies). The rxncon model files are available as Supplementary Model 1 (HOG model), Supplementary Model 2 (final pheromone model) or through download from <https://github.com/rxncon/models/> (YeastPheromoneModel.xls (the initial pheromone model)).

DATA AVAILABILITY

The models and code are freely available through the paper or public repositories. The rxncon software is open source, distributed under the IGPL licence, and can be installed from the python package index with "pip install rxncon". The code is also available downloaded from <https://github.com/rxncon/rxncon>. The rxncon model files are available as Supplementary Model 1 (HOG model), Supplementary Model 2 (final pheromone model), Supplementary Model 3 (covalent motif), Supplementary Model 4 (interaction motif) or through download from <https://github.com/rxncon/models/> (YeastPheromoneModel.xls (initial pheromone model)).

Received: 18 August 2018; Accepted: 20 November 2019;
Published online: 10 January 2020

REFERENCES

1. Fisher, J. & Henzinger, T. A. Executable cell biology. *Nat. Biotechnol.* **25**, 1239–1249 (2007).
2. Herrgard, M. J. et al. A consensus yeast metabolic network reconstruction obtained from a community approach to systems biology. *Nat. Biotechnol.* **26**, 1155–1160 (2008).
3. Thiele, I. et al. A community-driven global reconstruction of human metabolism. *Nat. Biotechnol.* **31**, 419–425 (2013).
4. Thiele, I. & Palsson, B. O. A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nat. Protoc.* **5**, 93–121 (2010).
5. Münzner, U., Lubitz, T., Klipp, E. & Krantz, M. In *Systems Biology* (eds Nielsen, J. & Hohmann, S.) 215–242 (Wiley, 2017).
6. Hlavacek, W. S. & Faeder, J. R. The complexity of cell signaling and the need for a new mechanics. *Sci. Signal* **2**, pe46 (2009).
7. Hlavacek, W. S., Faeder, J. R., Blinov, M. L., Perelson, A. S. & Goldstein, B. The complexity of complexes in signal transduction. *Biotechnol. Bioeng.* **84**, 783–794 (2003).

8. Blinov, M. L., Faeder, J. R., Goldstein, B. & Hlavacek, W. S. BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics* **20**, 3289–3291 (2004).
9. Danos, V., Feret, J., Fontana, W., Harmer, R. & Krivine, J. Rule-Based Modelling of Cellular Signalling. In *CONCUR 2007—Concurrency Theory: 18th International Conference, CONCUR 2007, Lisbon, Portugal, September 3–8, 2007. Proceedings* (eds Cairns, L. & Vasconcelos, V. T.) 17–41 (Springer, Berlin, Heidelberg, 2007).
10. Sneddon, M. W., Faeder, J. R. & Emonet, T. Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. *Nat. Methods* **8**, 177–183 (2011).
11. Machado, D. et al. Modeling formalisms in systems biology. *AMB Express* **1**, 45 (2011).
12. Abou-Jaoude, W. et al. Logical modeling and dynamical analysis of cellular networks. *Front Genet.* **7**, 94 (2016).
13. Handorf, T. & Klipp, E. Modeling mechanistic biological networks: an advanced Boolean approach. *Bioinformatics* **28**, 557–563 (2012).
14. Kolczyk, K., Samaga, R., Conzelmann, H., Mirschel, S. & Conradi, C. The Process-Interaction-Model: a common representation of rule-based and logical models allows studying signal transduction on different levels of detail. *BMC Bioinform.* **13**, 251 (2012).
15. Flottmann, M., Krause, F., Klipp, E. & Krantz, M. Reaction-contingency based bipartite Boolean modelling. *BMC Syst. Biol.* **7**, 58 (2013).
16. Romers, J. C. & Krantz, M. rxncon 2.0: a language for executable molecular systems biology. *bioRxiv*. <https://doi.org/10.1101/107136> (2017).
17. Münzner, U., Klipp, E. & Krantz, M. A comprehensive, mechanistically detailed, and executable model of the cell division cycle in *Saccharomyces cerevisiae*. *Nat. Commun.* **10**, 1308 (2019).
18. Mussel, C., Hopfensitz, M. & Kestler, H. A. BoolNet—An R package for generation, reconstruction and analysis of Boolean networks. *Bioinformatics* **26**, 1378–1380 (2010).
19. Klipp, E., Nordlander, B., Kruger, R., Gennemark, P. & Hohmann, S. Integrative model of the response of yeast to osmotic shock. *Nat. Biotechnol.* **23**, 975–982 (2005).
20. Thomson, T. M. *Yeast Pheromone Model*, <http://yeastpheromonemodel.org>.
21. Thomson, T. M. et al. Scaffold number in yeast signaling system sets tradeoff between system output and dynamic range. *Proc. Natl Acad. Sci. USA* **108**, 20265–20270 (2011).
22. Romers, J., Thieme, S., Münzner, U. & Krantz, M. in *Modeling Biomolecular Site Dynamics: Methods and Protocols* (ed. Hlavacek, W. S.) 71–118 (Springer, New York, 2019).
23. Saez-Rodriguez, J. et al. Discrete logic modelling as a means to link protein signalling networks with functional analysis of mammalian signal transduction. *Mol. Syst. Biol.* **5**, 331 (2009).
24. Fabregat, A. et al. The reactome pathway knowledgebase. *Nucleic Acids Res.* **44**, D481–D487 (2016).
25. Kawakami, E. et al. Network analyses based on comprehensive molecular interaction maps reveal robust control structures in yeast stress response pathways. *Npj Syst. Biol. Appl.* **2**, 15018 (2016).
26. Kuperstein, I. et al. Atlas of cancer signalling network: a systems biology resource for integrative analysis of cancer data with Google Maps. *Oncogenesis* **4**, e160 (2015).
27. Mori, T., Flottmann, M., Krantz, M., Akutsu, T. & Klipp, E. Stochastic simulation of Boolean rxncon models: towards quantitative analysis of large signaling networks. *BMC Syst. Biol.* **9**, 45 (2015).
28. Lubitz, T. et al. Network reconstruction and validation of the Snf1/AMPK pathway in baker's yeast based on a comprehensive literature review. *Npj Syst. Biol. Appl.* **1**, 15007 (2015).
29. Tiger, C. F. et al. A framework for mapping, visualisation and automatic model creation of signal-transduction networks. *Mol. Syst. Biol.* **8**, 578 (2012).
30. Creamer, M. S. et al. Specification, annotation, visualization and simulation of a large rule-based model for ERBB receptor signaling. *BMC Syst. Biol.* **6**, 107 (2012).
31. Srinivasan, S., Cluett, W. R. & Mahadevan, R. Constructing kinetic models of metabolism at genome-scales: a review. *Biotechnol. J.* **10**, 1345–1359 (2015).

ACKNOWLEDGEMENTS

This work was supported by the German Federal Ministry of Education and Research via e:Bio Cellemental (FKZ0316193, to MK).

AUTHOR CONTRIBUTIONS

J.R., S.T. and M.K. designed the method. J.R. and S.T. implemented the method, and contributed equally to the work. U.M. validated the method. All authors contributed to and approved of the final manuscript.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Supplementary information is available for this paper at <https://doi.org/10.1038/s41540-019-0120-5>.

Correspondence and requests for materials should be addressed to M.K.

Reprints and permission information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2020